API PIX AUTOMÁTICO
MANUAL TÉCNICO

SUMÁRIO

PARTE 4

Objetivo

O objetivo deste manual é orientar sobre como proceder com a integração da API Pix Automático, detalhando os processos de autenticação, autorização de aplicações server-to-server e os aspectos negociais para uso das funcionalidades.

Parte 1

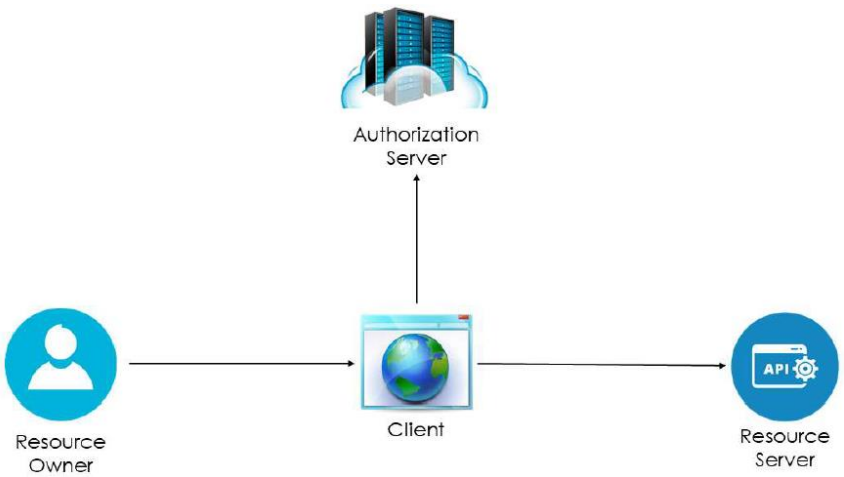
1.REQUISITOS OBRIGATÓRIOS E RECOMENDAÇÕES DE SEGURANÇA

1.1Protocolos:

A CAIXA implementa um protocolo de autenticação e autorização no padrão OpenID Connect (OIDC) baseado no framework OAuth 2.0.

O OIDC faz uso do conjunto de padrões JSON Web Token (JWT) que definem um formato JSON de token de identidade e formas de assinar e criptografar digitalmente esses dados de maneira compacta e amigável.

Em uma arquitetura com segurança baseado em OAuth2, a autenticação é o processo em que um usuário, pessoa ou serviço apresenta suas credenciais junto a um Servidor de Autorização (AS) e obtém um token de acesso com o qual a aplicação faz a requisição de um recurso protegido



1.2 Token JWT (JSON Web Token)

O JSON Web Token (JWT) é uma forma compacta e segura de representar declarações a serem transferidas entre duas partes.

As declarações em um JWT são codificadas como um objeto JSON que é usado como payload de uma estrutura JSON Web Signature (JWS) ou como texto simples de uma estrutura JSON Web Encryption (JWE), permitindo que as declarações sejam assinadas digitalmente ou protegidas por integridade com um Código de Autenticação de Mensagem (MAC) e / ou criptografia.

Os tokens obtidos por meio do protocolo oAuth2 seguem o padrão JWT, o qual foi desenhado para ser compacto (comparado a outros padrões de token) e URL-Safe, tornando-o ideal para ser usado com o protocolo HTTP.

A estrutura de um JWT é composta por 3 partes: cabeçalho, conteúdo (payload) e assinatura, unidos cada um por ponto (.) conforme o exemplo abaixo:

eyJhbG[...]V1QifQ.e2lzcz[...]QifQ0K.g5g6HN[...]j2Lsuw

1.2.1.Cabeçalho

O cabeçalho identifica o tipo de token e o algoritmo utilizado para assinar o token. Os dados são convertidos para Base64.

Decodificado:

{"alg":"RS256","typ":"JWT"}

Codificado:

eyJhbGciOiIgUlMyNTYiLCJ0eXAiOiJKV1QifQ

1.2.2.Conteúdo (Payload)

O conteúdo de um token no padrão JWT possui atributos (claims) necessários para a utilização do token. Para o protocolo OAuth2, o token de acesso possui os seguintes atributos relevantes:

iss (Issuer)

Identifica o emissor do token, geralmente contém a URL do Servidor de Autorização.

sub (Subject)

Identifica o usuário para o qual foi gerado o token.

aud (Audience)

Identifica a aplicação por meio da qual foi solicitado o token.

exp (Expiration Time)

Identifica o tempo de expiração do token.

iat (Issuet At)

Identifica o momento em que o token foi emitido.

Decodificado

```
{
 "iss": "http://login.caixa.gov.br/auth/realms/internet",
 "sub": "ae9fd4e7-ed15-494f-ba0b-660d70d1bf5d",
 "aud": "application_id",
 "exp": 1492028993,
 "iat": 1492028933,
 "clientId": "application_id",
 "preferred_username ": "user_id"
}
```

Codificado:

```
Ew0KaXNzOiAiaHR0cDovL2xvZ2luLmNhaXhhL2F1dGgvcmVhbG1zL2ludHJhbmV0IiwNCnN1YjogImFlOWZkNGU3LWVkMTUtNDk0Zi1iYTBiLTY2MGQ3MGQxYmY1ZCIsDQphdWQ6ICJhcHBsaWNhdGlvbl9pZCIsDQpleHA6IDE0OTIwMjg5OTMsDQppYXQ6IDE0OTIwMjg5MzMsDQpjbGllbnRJZDogImFwcGxpY2F0aW9uX2lkIiwNCnByZWZlcnJlZF91c2VybmFtZTogInVzZXJfaWQiDQp9
```

1.2.3 Assinatura

A assinatura é calculada com base no cabeçalho e no conteúdo codificados para Base64, separados por ponto (.) e utilizando algoritmo identificado no cabeçalho com a chave privada do Servidor de Autorização.

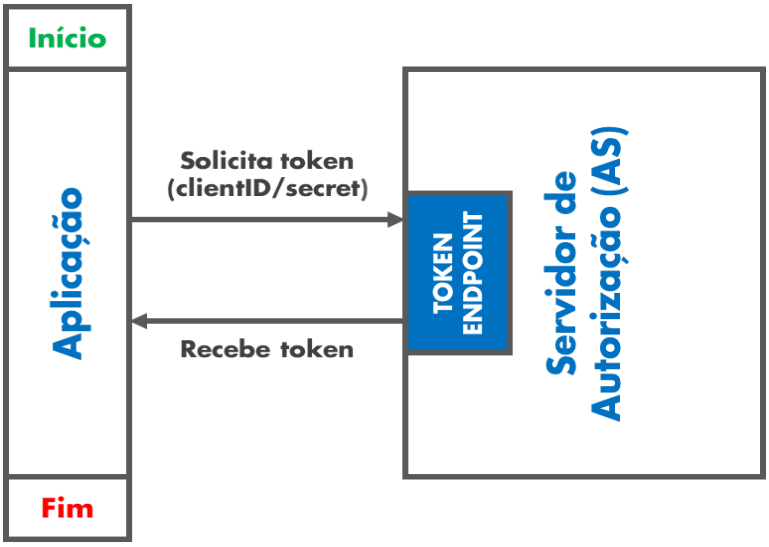
```
RSA-SHA256(chave, (base64(cabeçalho).base64(conteúdo)))
```

1.2.4 Grant Client Credentials

O Grant Client Credentials é indicado para os casos em que a aplicação necessita acessar um recurso protegido em nome próprio (usuário de serviço) sem o envolvimento de usuário pessoa.

No Grant Client Credentials a aplicação interage apenas com o serviço de obtenção de Token do Servidor de Autorização (Token Endpoint), apresentando somente as suas próprias credenciais. O Token Endpoint devolve tanto um Token de acesso quanto um Token de renovação para a aplicação.

A figura a seguir ilustra o fluxo Client Credentials:



1.2.5 Acessando o Token Endpoint

O Token Endpoint é um serviço REST por meio do qual a aplicação vai interagir com o Servidor de Autorização para obter ou renovar os tokens.

O serviço implementa o método POST para receber os atributos necessários para requisição dos tokens. Os atributos necessários são os seguintes:

grant_type

Especifica o grant que está sendo utilizado (Authorization Code ou Client Credentials).

client_id

Especifica a aplicação que está solicitando o token.

client_secret

Especifica a credencial (senha) da aplicação.

O usuário recebedor deve acionar a CAIXA, por meio do seu gerente de relacionamento, para obter a APIKey, o client id e o client secret.

1.2.6 Grant Authorization Code (grant_type=authorization_code)

redirect_uri

Especifica a URL utilizada na requisição ao Authorization Endpoint.

code

Especifica o Authorization Code devolvido pelo Authorization Endpoint após a autenticação positiva.

1.2.7 Grant Client Credentials (grant_type=client_credentials)

scope

Especifica o escopo da autorização que está sendo solicitada. Identifique, na descrição do Swagger da API, o scope necessário para acesso ao recurso.

O Token Endpoint também é utilizado para renovar o token de acesso por meio do token de renovação. Para esse cenário, os atributos necessários são os seguintes:

grant_type

Especifica o grant que está sendo utilizado. Nesse caso, refresh_token.

cliente_id

Especifica a aplicação que está solicitando o token.

cliente_secret

Especifica a credencial (senha) da aplicação.

refresh_token

Especifica o token de renovação recebido pela aplicação.

1.2.8 Tempo de vida

É necessário a utilização do token durante todo o seu tempo de vida (validade). Após a expiração do tempo de vida (exp) do token de acesso (access_token), é necessária a utilização de refresh token.

O tempo de validade do token é fixo conforme descrito na geração do mesmo independente de virada de data.

O tempo de validade do access_token é de 15 min (o tempo de expiração vem expresso em segundos).

O tempo de validade do refresh token é de 30 min e o tempo máximo da sessão 24hs.

É vedado o uso de processos BATCH de rajada (em lote) nas chamadas de API, tendo em vista os eventuais impactos na infraestrutura do SSO CAIXA e das API. Também é vedado o uso de API para listagem de registros da base sem o consentimento do usuário final.

Recomendamos verificar o preenchimento do parâmetro http_user_agent do header da requisição evitando utilizar valores genéricos que possam ser confundidos com bots, causando assim bloqueio pela

ferramenta de segurança anti-bot da CAIXA.

Exemplos de requisições para o Token Endpoint

- authorization code

```
POST https://login.caixa.gov.br/auth/realms/internet/protocol/openid-connect/token
Content-Type: application/x-www-form-urlencoded
grant_type=authorization_code
&code=SplxlOBeZQQYbYS6WxSbIA
&redirect_uri=https%3A%2F%2Faplicacao.teste%2Fapos_login
&client_id=id_aplicacao
&client_secret=7Fjfp0ZBr1KtDRbnfVdmIw
```

- client credentials

```
POST https://login.caixa.gov.br/auth/realms/internet/protocol/openid-connect/token
Content-Type: application/x-www-form-urlencoded
grant_type=client_credentials
&client_id=id_aplicacao
&client_secret=7Fjfp0ZBr1KtDRbnfVdmIw
```

- refresh token

```
POST https://login.caixa.gov.br/auth/realms/internet/protocol/openid-connect/token
Content-Type: application/x-www-form-urlencoded
grant_type=refresh_token
&refresh_token=tGzv3JOkF0X...G5Qx2TlKWIA
&client_id=id_aplicacao
&client_secret=7Fjfp0ZBr1KtDRbnfVdmIw
```

A resposta do Token Endpoint será uma mensagem no formato JSON contendo os tokens gerados pelo Servidor de Autorização

Parte 2

2. Integração

2.1 Hosts do Servidor de Autorização

Além do ambiente de produção, a CAIXA provê um ambiente para teste de consumo das suas APIs, conhecido como SandBox. As URLs do Authorization Endpoint e do Token Endpoint de cada ambiente estão descritas abaixo:

- SandBox

https://logindes.caixa.gov.br/auth/realms/internet/protocol/openid-connect/token

- Produção

https://loginservicos.caixa.gov.br/auth/realms/internet/protocol/openid-connect/token

2.2 Procedimentos de Testes de Homologação

A Caixa exige que a empresa realize testes obrigatórios antes de liberar produção:

2.2.1. Configuração do ambiente sandbox:

- Obter credenciais temporárias.

- Configurar endpoints de teste (ex.: https://sandbox.caixa.gov.br/pix-automatico/v1/...).

2.2.2 Testes funcionais

- Criar consentimento: enviar dados do pagador e validar retorno.

- Agendar débito: simular recorrência (ex.: mensal).

- Consultar status: verificar liquidação simulada.

- Cancelar cobrança: testar rejeição após prazo limite.

2.2.3 Validação de regras de negócio

- Testar limites de valor, periodicidade, e tratamento de erros (HTTP 4xx, 5xx).

- Confirmar aderência ao Manual de Experiência do Usuário (ex.: telas de autorização).

2.2.4. Testes de segurança

- Autenticação via OAuth.

- Certificados válidos.

- Logs e auditoria.

2.3 Fluxo de Autenticação do Cliente

A autenticação é baseada no padrão OAuth 2.0 Client Credentials, garantindo segurança e controle de acesso.

1.Credenciais fornecidas pela Caixa

Após contratar o convênio Pix Automático, a empresa recebe:client_id e .client_secret

Essas credenciais são usadas para gerar o token de acesso.

2.Solicitação do Token

Endpoint: POST /oauth2/token

Headers: Content-Type: application/x-www-form-urlencoded

Body: grant_type=client_credentials

Exemplo resposta:

```
{¶
··"access_token":·"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",¶
··"token_type":·"Bearer",¶
··"expires_in":·3600¶
}¶
```

Esse access_token deve ser incluído no header Authorization: Bearer {token} em todas as chamadas subsequentes

3.Renovação do Token

O token expira (geralmente em 1 hora).

A aplicação deve implementar lógica para renovar automaticamente antes da expiração.

2.4 Migração para Produção

Após aprovação:

- A Caixa libera credenciais definitivas.
- A empresa aponta para os endpoints de produção
- É necessário configurar webhooks para receber notificações de liquidação em tempo real.

Parte 3

3.Jornadas

3.1 Jornada 1: Push

PJ envia pedido de autorização por arquivo, API ou canal de autoatendimento e o pagador recebe PUSH/SMS para confirmação

Para criar uma solicitação de autorização via Jornada 1, inicia acionando o serviço REC (Post), informando os dados para obtenção do ID da recorrência, conforme exemplo disponível no Swagger UI

```
{
  "vinculo": {
    "contrato": "63100862",
    "devedor": {
      "cpf": "45164632481",
      "nome": "Fulano de Tal"
    },
    "objeto": "Serviço de Streamming de Música."
  },
  "calendario": {
    "dataFinal": "2025-04-01",
    "dataInicial": "2024-04-01",
    "periodicidade": "MENSAL"
  },
  "valor": {
    "valorRec": "35.00"
  },
  "politicaRetentativa": "NAO_PERMITE",
  "loc": 108,
  "ativacao": {
    "dadosJornada": {
      "txid": "33beb661beda44a8928fef47dbeb2dc5"
    }
  }
}
```

Após receber o ID, aciona o serviço SolicRec (Post) para criação da solicitação de confirmação da recorrência, conforme exemplo disponível no https://bacen.github.io/pix-api/#/SolicRec/post_solicrec

```
{
  "idRec": "RN12345678202401157782544 5612",
  "calendario": {
    "dataExpiracaoSolicitacao": "2023-12-20T12:17:11.926Z"
  },
  "destinatario": {
    "agencia": "2569",
    "conta": "550689",
    "cpf": "15231470190",
    "ispbParticipante": "91193552"
  }
```

```
}
```

3.2 Jornada 2, 3 e 4: QR Code

O serviço a ser utilizado para a geração de QR code dependerá da Modalidade QR Code selecionada pelo usuário (Jornada 2, 3 ou 4), conforme segue:

Jornada 2: AUTORIZAÇÃO:

PJ gera e disponibiliza QR Code e o pagador lê QR Code para autorizar.

utilizar o serviço Swagger UI para criação da Location do payload

```
{
  "id": 12069,
  "location": "pix.example.com/qr/v2/rec/2353c790eefb11eaadc10242ac120002",
  "criacao": "2023-12-20T12:38:28.774Z"
}
```

e o Swagger UI informando a ID da Location

```
{
  "vinculo": {
    "contrato": "63100862",
    "devedor": {
      "cpf": "45164632481",
      "nome": "Fulano de Tal"
    },
    "objeto": "Serviço de Streamming de Música."
  },
  "calendario": {
    "dataFinal": "2025-04-01",
    "dataInicial": "2024-04-01",
    "periodicidade": "MENSAL"
  },
  "valor": {
    "valorRec": "35.00"
  },
  "politicaRetentativa": "NAO_PERMITE",
  "loc": 108,
  "ativacao": {
    "dadosJornada": {
      "txid": "33beb661beda44a8928fef47dbeb2dc5"
    }
  }
```

Com as informações obtidas, deve gerar o QR Code para disponibilizar em tela (o sistema cliente é responsável por montar o QR Code para exibição).

Jornada 3: PAGAMENTO + AUTORIZAÇÃO:

PJ gera e envia um único QR Code contendo pagamento imediato e pedido de autorização e o pagador lê QR Code para fazer o pagamento e autorizar

Primeiramente, utilizar o serviço Post/PayloadLocation Swagger UI para criar a Location (ID).

```
{
  "tipoCob": "cob"
}
```

Depois, realiza Post/Cob Swagger UI informando o ID Location para geração do pagamento imediato (TXid).

```
{
  "calendario": {
    "expiracao": 3600
  },
  "devedor": {
    "cnpj": "12345678000195",
    "nome": "Empresa de Serviços SA"
  },
  "valor": {
    "original": "37.00",
    "modalidadeAlteracao": 1
  },
  "chave": "7d9f0335-8dcc-4054-9bf9-0dbd61d36906",
  "solicitacaoPagador": "Serviço realizado.",
  "infoAdicionais": [
    {
      "nome": "Campo 1",
      "valor": "Informação Adicional1 do PSP-Recebedor"
    },
    {
      "nome": "Campo 2",
      "valor": "Informação Adicional2 do PSP-Recebedor"
    }
  ]
}
```

Na sequência, cria a recorrência no Swagger UI

```
{
  "vinculo": {
    "contrato": "63100862",
    "devedor": {
      "cpf": "45164632481",
      "nome": "Fulano de Tal"
    },
    "objeto": "Serviço de Streamming de Música."
  },
  "calendario": {
    "dataFinal": "2025-04-01",
    "dataInicial": "2024-04-01",
    "periodicidade": "MENSAL"
  },
  "valor": {
    "valorRec": "35.00"
  },
  "politicaRetentativa": "NAO_PERMITE",
  "loc": 108,
  "ativacao": {
    "dadosJornada": {
      "txid": "33beb661beda44a8928fef47dbeb2dc5"
    }
  }
}
```

e então, realiza chamada no Swagger UI com o TXid e IdRec obtidos no passo anterior

```
{
  "idRec": "RR12345678202401l5abcdefghijk",
  "infoAdicional": "Serviços de Streamming de Música e Filmes.",
  "calendario": {
    "dataDeVencimento": "2024-04-15"
  },
  "valor": {
    "original": "106.07"
  },
  "ajusteDiaUtil": true,
  "devedor": {
    "cep": "89256-140",
    "cidade": "Uberlândia",
    "email": "sebastiao.tavares@mail.com",
    "logradouro": "Alameda Franco 1056",
    "uf": "MG"
  },
  "recebedor": {
    "agencia": "9708",
    "conta": "012682",
    "tipoConta": "CORRENTE"
  }
}
```

Com as informações obtidas, gerar o QR Code para disponibilizar ao pagador (o sistema cliente é responsável por montar o QR Code para exibição).

Jornada 4: PAGAMENTO + OFERTA DE AUTORIZAÇÃO:

PJ gera e envia um único QR Code contendo pagamento imediato e uma oferta, o pagador lê QR Code para fazer o pagamento e ao final recebe a oferta para autorizar

Primeiramente, utilizar o serviço Post/PayloadLocation Swagger UI para criar a Location (ID).

```
{
  "tipoCob": "cob"
}
```

Depois, realiza Post/Cob Swagger UI informando o ID Location para geração do pagamento imediato (TXid).

```
{
  "calendario": {
    "expiracao": 3600
  },
  "devedor": {
    "cnpj": "12345678000195",
    "nome": "Empresa de Serviços SA"
  },
  "valor": {
    "original": "37.00",
    "modalidadeAlteracao": 1
  },
  "chave": "7d9f0335-8dcc-4054-9bf9-0dbd61d36906",
  "solicitacaoPagador": "Serviço realizado.",
  "infoAdicionais": [
    {
```

```
      "nome": "Campo 1",
      "valor": "Informação Adicional1 do PSP-Recebedor"
    },
    {
      "nome": "Campo 2",
      "valor": "Informação Adicional2 do PSP-Recebedor"
    }
  ]
}
```

Por fim, utilizar o serviço Swagger UI para criação da Location do payload

```
{
  "id": 12069,
  "location": "pix.example.com/qr/v2/rec/2353c790eefb11eaadc10242ac120002",
  "criacao": "2023-12-20T12:38:28.774Z"
}
```

e o Swagger UI informando o ultimo ID da LocationRec.

```
{
  "vinculo": {
    "contrato": "63100862",
    "devedor": {
      "cpf": "45164632481",
      "nome": "Fulano de Tal"
    },
    "objeto": "Serviço de Streamming de Música."
  },
  "calendario": {
    "dataFinal": "2025-04-01",
    "dataInicial": "2024-04-01",
    "periodicidade": "MENSAL"
  },
  "valor": {
    "valorRec": "35.00"
  },
  "politicaRetentativa": "NAO_PERMITE",
  "loc": 108,
  "ativacao": {
    "dadosJornada": {
      "txid": "33beb661beda44a8928fef47dbeb2dc5"
    }
  }
```

Com as informações obtidas, deve gerar o QR Code Composto para disponibilizar em tela (o sistema cliente é responsável por montar o QR Code para exibição), lembrando que o QR Code composto carrega as duas URL vinculadas aos seus respectivos ID.

Parte 4

4.Endopoint API Caixa

4.1 RECORRENCIA (/rec e /solicrec)

4.1.1 Criar recorrência de Pix Automático Swagger UI

Endpoint: POST /rec

Exemplo:


```
{
  "vinculo": {
    "contrato": "63100862",
    "devedor": {
      "cpf": "45164632481",
      "nome": "Fulano de Tal"
    },
    "objeto": "Serviço de Streamming de Música."
  },
  "calendario": {
    "dataFinal": "2025-04-01",
    "dataInicial": "2024-04-01",
    "periodicidade": "MENSAL"
  },
  "valor": {
    "valorRec": "35.00"
  },
  "politicaRetentativa": "NAO_PERMITE",
  "loc": 108,
  "ativacao": {
    "dadosJornada": {
      "txid": "33beb661beda44a8928fef47dbeb2dc5"
    }
  }
}
```


4.1.2. Consultar recorrência Swagger UI

Endpoint: GET /rec/{idRec}

Exemplo:


```
{
  "idRec": "RN12345678202401l5abcdefghijk",
  "status": "APROVADA",
  "valor": {
    "valorRec": "300.00"
  },
  "vinculo": {
```

```
    "contrato": "98625023",
    "devedor": {
      "cpf": "87734514122",
      "nome": "Fulano de Tal"
    },
    "objeto": "Serviços de Gestão de Imóveis"
  },
  "calendario": {
    "dataFinal": "2028-09-01",
    "dataInicial": "2024-02-01",
    "periodicidade": "MENSAL"
  },
  "politicaRetentativa": "NAO_PERMITE",
  "loc": {
    "criacao": "2023-12-19T12:28:05.230Z",
    "id": 5100,
    "location": "pix.example.com/qr/v2/rec/2353c790eefb11eaadc10242ac120002",
    "idRec": "RN1234567820240115abcdefghijk"
  },
  "pagador": {
    "codMun": "2673833",
    "cpf": "75633122216",
    "ispbParticipante": "81102623"
  },
  "recebedor": {
    "cnpj": "92221288310574",
    "nome": "Imobiliária Bom Sucesso"
  },
  "atualizacao": [
    {
      "data": "2024-01-03T08:30:02.050Z",
      "nome": "CRIADA"
    },
    {
      "data": "2024-01-04T09:40:42.210Z",
      "nome": "APROVADA"
    }
  ],
  "dadosQR": {
    "jornada": "JORNADA_2",
    "pixCopiaECola": "00020126180014br.gov.bcb.pix5204000053039865802BR5913Fulano de Tal6008BRA
SILIA62070503***80800014br.gov.bcb.pix2558pix.example.com/qr/v2/rec/2353c790eefb11eaadc10242ac1
20002630462C9"
  }
}
```


4.1.3. Criar solicitação de confirmação de recorrência Swagger UI

Endpoint: POST /solicrec

Exemplo:


```
{
  "idRec": "RN12345678202401157782544 5612",
  "calendario": {
    "dataExpiracaoSolicitacao": "2023-12-20T12:17:11.926Z"
  },
  "destinatario": {
    "agencia": "2569",
```

```
    "conta": "550689",
    "cpf": "15231470190",
    "ispbParticipante": "91193552"
  }
}
```


4.1.4. Consultar solicitação de recorrência Swagger UI

Endpoint: GET /solicrec/{idSolicRec

Exemplo:


```
{
  "idSolicRec": "SC87645678202402157782544 5312",
  "idRec": "RN12345678202401157782544 5612",
  "calendario": {
    "dataExpiracaoSolicitacao": "2023-12-20T12:17:11.926Z"
  },
  "status": "CRIADA",
  "destinatario": {
    "agencia": "2569",
    "conta": "550689",
    "cpf": "15231470190",
    "ispbParticipante": "91193552"
  },
  "atualizacao": [
    {
      "data": "2023-12-20T12:18:18.618Z",
      "status": "CRIADA"
    }
  ],
  "recPayload": {
    "idRec": "RN12345678202401157782544 5612",
    "vinculo": {
      "contrato": "561238008",
      "devedor": {
        "cpf": "15231470190",
        "nome": "Fulano de Tal"
      },
      "objeto": "Serviços de Telecomunicações"
    },
    "calendario": {
      "dataFinal": "2023-12-01",
      "dataInicial": "2024-04-01",
      "periodicidade": "MENSAL"
    },
    "recebedor": {
      "cnpj": "94370926517368",
      "nome": "Empresa de Serviços SA"
    },
    "valor": {
      "valorRec": "1200.09"
    },
    "atualizacao": [
      {
        "data": "2023-12-15T08:30:07.115Z",
        "status": "CRIADA"
      }
    ]
  }
```

4.2 COBRANÇA (/cob,/cobv,/cobr)

Representa cada uma das cobranças geradas por meio da API Pix, a fim de permitir que o usuário pagador efetue um pagamento identificado para o usuário recebedor. A cobrança é caracterizada por um conjunto de informações que são utilizadas para que o usuário pagador execute um pagamento por meio do Pix, geralmente, em função de acordo comercial entre o usuário pagador e o usuário recebedor, sem se confundir com o pagamento Pix em si. A cobrança se subdivide em duas espécies: cobranças para pagamento imediato (/cob) e cobranças para pagamento com vencimento (/cobv). Status da cobrança:

ATIVA: indica que a cobrança foi gerada e pronta para ser paga;

CONCLUÍDA: indica que a cobrança já foi paga e, por conseguinte, não pode acolher outro pagamento;

REMOVIDO_PELO_USUARIO_RECEBEDOR: indica que o usuário recebedor solicitou a remoção da cobrança;

REMOVIDO_PELO_PSP: indica que o PSP Recebedor solicitou a remoção da cobrança.

4.2.1. Criar cobrança imediata: Swagger UI

Endopoint: POST/cob

Exemplo:

```
{
  "calendario": {
    "expiracao": 3600
  },
  "devedor": {
    "cnpj": "12345678000195",
    "nome": "Empresa de Serviços SA"
  },
  "valor": {
    "original": "37.00",
    "modalidadeAlteracao": 1
  },
  "chave": "7d9f0335-8dcc-4054-9bf9-0dbd61d36906",
  "solicitacaoPagador": "Serviço realizado.",
  "infoAdicionais": [
    {
      "nome": "Campo 1",
      "valor": "Informação Adicional1 do PSP-Recebedor"
    },
    {
      "nome": "Campo 2",
      "valor": "Informação Adicional2 do PSP-Recebedor"
    }
  ]
}
```

4.2.2 Consultar cobrança imediata Swagger UI

Endpoint para consultar cobranças imediatas através de parâmetros como início, fim, cpf, cnpj e status.

Endopoint: GET/cob

Exemplo:


```
{
  "parametros": {
    "inicio": "2020-04-01T00:00:00Z",
    "fim": "2020-04-02T10:00:00Z",
    "paginacao": {
      "paginaAtual": 0,
      "itensPorPagina": 100,
      "quantidadeDePaginas": 1,
      "quantidadeTotalDeItens": 2
    }
  },
  "cobs": [
    {
      "$ref": "openapi.yaml#/components/examples/cobResponse1/value"
    },
    {
      "$ref": "openapi.yaml#/components/examples/cobResponse2/value"
    },
    {
      "$ref": "openapi.yaml#/components/examples/cobResponse5/value"
    },
    {
      "$ref": "openapi.yaml#/components/examples/cobResponse6/value"
    },
    {
      "$ref": "openapi.yaml#/components/examples/cobResponse7/value"
    }
  ]
}
```


4.2.3 Criar uma cobrança com vencimento. Swagger UI

Endopoint: PUT/cobv/{txid}

Exemplo:


```
{
  "calendario": {
    "dataDeVencimento": "2020-12-31",
    "validadeAposVencimento": 30
  },
  "loc": {
    "id": 789
  },
  "devedor": {
    "logradouro": "Alameda Souza, Numero 80, Bairro Braz",
    "cidade": "Recife",
    "uf": "PE",
    "cep": "70011750",
    "cpf": "12345678909",
    "nome": "Francisco da Silva"
```

```
  },
  "valor": {
    "original": "123.45",
    "multa": {
      "modalidade": "2",
      "valorPerc": "15.00"
    },
    "juros": {
      "modalidade": "2",
      "valorPerc": "2.00"
    },
    "desconto": {
      "modalidade": "1",
      "descontoDataFixa": [
        {
          "data": "2020-11-30",
          "valorPerc": "30.00"
        }
      ]
    }
  },
  "chave": "5f84a4c5-c5cb-4599-9f13-7eb4d419dacc",
  "solicitacaoPagador": "Cobrança dos serviços prestados."
}
```


4.2.4 Consultar uma cobrança com vencimento através de um determinado txid. Swagger UI

Endopoint: GET/cobv/{txid}

Exemplo:


```
{
  "calendario": {
    "criacao": "2020-09-09T20:15:00.358Z",
    "dataDeVencimento": "2020-12-31",
    "validadeAposVencimento": 30
  },
  "txid": "7978c0c97ea847e78e8849634473c1f1",
  "revisao": 0,
  "loc": {
    "id": 789,
    "location": "pix.example.com/qr/c2/cobv/9d36b84fc70b478fb95c12729b90ca25",
    "tipoCob": "cobv"
  },
  "status": "ATIVA",
  "devedor": {
    "logradouro": "Alameda Souza, Numero 80, Bairro Braz",
    "cidade": "Recife",
    "uf": "PE",
    "cep": "70011750",
    "cpf": "12345678909",
    "nome": "Francisco da Silva"
  },
  "recebedor": {
    "logradouro": "Rua 15 Numero 1200, Bairro São Luiz",
    "cidade": "São Paulo",
    "uf": "SP",
    "cep": "70800100",
    "cnpj": "56989000019533",
    "nome": "Empresa de Logística SA"
```

```
  },
  "valor": {
    "original": "123.45"
  },
  "chave": "5f84a4c5-c5cb-4599-9f13-7eb4d419dacc",
  "solicitacaoPagador": "Cobrança dos serviços prestados."
}
```

4.2.5 Criar uma cobrança recorrente Swagger UI

Endoipoint para criar cobrança reccorente,neste caso, o txid deve ser definido pelo PSP

Endopoint:POST/cobr

Exemplo:

```
{
  "idRec": "RR1234567820240115abcdefghijk",
  "infoAdicional": "Serviços de Streamming de Música e Filmes.",
  "calendario": {
    "dataDeVencimento": "2024-04-15"
  },
  "valor": {
    "original": "106.07"
  },
  "ajusteDiaUtil": true,
  "devedor": {
    "cep": "89256-140",
    "cidade": "Uberlândia",
    "email": "sebastiao.tavares@mail.com",
    "logradouro": "Alameda Franco 1056",
    "uf": "MG"
  },
  "recebedor": {
    "agencia": "9708",
    "conta": "012682",
    "tipoConta": "CORRENTE"
  }
}
```

4.2.6. Consultar cobranças recorrentes

Endpoint para consultar cobranças recorrentes através de parâmetros como início, fim, idRec, cpf, cnpj, status e convênio Swagger UI

Endopoint:GET/cobr

Exemplo:

```
{
  "parametros": {
    "inicio": "2024-04-01T00:00:00Z",
    "fim": "2024-12-01T23:59:59Z",
    "paginacao": {
      "paginaAtual": 0,
      "itensPorPagina": 100,
      "quantidadeDePaginas": 1,
      "quantidadeTotalDeItens": 1
```

```
    }
  },
  "cobsr": [
    {
      "idRec": "RR12345678202406199900056635 4",
      "txid": "7f733863543b4a16b516d839bd4bc34e",
      "calendario": {
        "criacao": "2024-05-20",
        "dataDeVencimento": "2024-06-20"
      },
      "valor": {
        "original": "50.33"
      },
      "status": "ATIVA",
      "ajusteDiaUtil": false,
      "politicaRetentativa": "PERMITE_3R_7D",
      "devedor": {
        "cep": "63259-740",
        "cidade": "Campinas",
        "email": "beltrano.silva@mail.com",
        "logradouro": "Rua Gonçalves Dias 605",
        "uf": "SP"
      },
      "recebedor": {
        "conta": "997182",
        "tipoConta": "CORRENTE"
      },
      "tentativas": [
        {
          "dataLiquidacao": "2024-06-20",
          "tipo": "AGND",
          "status": "AGENDADA",
          "endToEndId": "E12345678202406201221abcdef12345",
          "atualizacao": [
            {
              "data": "2024-05-21T10:40:16.730Z",
              "status": "SOLICITADA"
            },
            {
              "data": "2024-05-21T17:08:00.520Z",
              "status": "AGENDADA"
            }
          ]
        }
      ],
      "atualizacao": [
        {
          "data": "2024-05-20T14:47:29.470Z",
          "status": "CRIADA"
        },
        {
          "data": "2024-05-21T10:18:20.120Z",
          "status": "ATIVA"
        }
      ]
    }
  ]
}
```

4.3 Tratamento de Erros

Descrição dos possíveis erros que podem ocorrer durante o uso dos endpoints para o adequado tratamento, seguindo a documentação do PIX BACEN Swagger UI

A API Pix retorna códigos de status HTTP para indicar sucesso ou falhas das requisições, são eles:

• Códigos 2xx indicam sucesso;

• Códigos 4xx indicam falhas causadas pelas informações enviadas pelo cliente ou pelo estado atual das antedates e;

• Códigos 5xx indicam problemas no serviço no lado da API Pix. As respostas de erro incluem no corpo detalhes do erro seguindo o schema da RFC 7807.

O campo type identifica o tipo de erro e na API Pix segue o padrão: https://pix.bcb.gov.br/api/v2/error/

O padrão acima listado, referente ao campo type, não consiste, necessariamente, em uma URL que apresentará uma página web válida, ou um endpoint válido, embora possa, futuramente, ser exatamente o caso.

O objetivo primário é apenas e tão somente identificar o tipo de erro. Convém reforçar que a API Pix contempla uma lista de produtos e respectivas funcionalidades ofertadas pelo PSP recebedor.

Cabe à relação contratual com cada usuário recebedor a concessão da totalidade ou de um subconjunto de acessos relacionados aos produtos ofertados.

Por exemplo, o usuário recebedor, ao acessar uma funcionalidade não contemplada no seu escopo contratual, receberá o erro geral AcessoNegado descrito na próxima seção. Abaixo estão listados os tipos de erro e possíveis violações da API Pix.

4.3.1 GERAIS

Esta seção reúne erros que poderiam ser retornados por quaisquer endpoints listados na API Pix.

RequisicaoInvalida:

• Significado: Requisição inválida.

• HTTP Status Code: 400 Bad Request.

AcessoNegado:

• Significado: Requisição de participante autenticado que viola alguma regra de autorização.

• HTTP Status Code: 403Forbidden.

NaoEncontrado:

• Significado: Entidade não encontrada. 38503 v002 micro 28

• HTTP Status Code: 404 Not Found.

PermanentementeRemovido:

• Significado: Indica que a entidade existia, mas foi permanentemente removida.

• HTTP Status Code: 410 Gone.

ErroInternoDoServidor:

• Significado: Condição inesperada ao processar requisição.

• HTTP Status Code: 500 Internal Server Error.

ServicoIndisponivel:

• Significado: Serviço não está disponível no momento. Serviço solicitado pode estar em manutenção ou fora da janela de funcionamento.

• HTTP Status Code: 503 Service Unavailable.

IndisponibilidadePorTempoEsgotado:

• Significado: Indica que o serviço demorou além do esperado para retornar

• HTTP Status Code: 504 Gateway Timeout.

4.3.2 Rec

Esta seção reúne erros retornados pelos endpoints organizados sob a tag Rec.

Esses erros indicam problemas no gerenciamento de uma recorrência.

RecNaoEncontrada

• Significado: Recorrência não encontrada para o idRec informado.

• HTTP Status Code: 404.

• endpoints: [GET|PATCH] /rec/{idRec}.

RecOperacaoInvalida

• Significado: a requisição que busca alterar ou criar uma recorrência não respeita o schema ou está semanticamente errada.

• HTTP Status Code: 400.

• endpoints: POST /rec e PATCH /rec/{idRec}.

Violações para o endpoint POST /rec:

• O objeto rec.vinculo não respeita o schema.

• O campo rec.calendario.dataInicial é anterior à data de criação da recorrência

• O campo rec.calendario.dataFinal é anterior ao campo rec.calendario.dataInicial. • O campo rec.calendario.periodicidade não respeita o schema.

• O objeto rec.valor não respeita o schema.

• O campo rec.valor.valorRec não respeita o schema. 38503 v002 micro 30 • O campo rec.valor.valorMinimoRecebedor não respeita o schema.

• Ambos os campos rec.valor.valorRec e rec.valor.valorMinimoRecebedor estão preenchidos.

• O objeto rec.recebedor não respeita o schema.

• O campo rec.politicaRetentativa não respeita o schema.

• O location referenciado por rec.loc inexiste.

• O location referenciado por rec.loc já está sendo utilizado por outra recorrência.

• O valor do campo rec.recebedor.convenio não é aceito pelo PSP Recebedor.

Violações para o endpoint PATCH /rec/{idRec}:

• O campo rec.calendario.dataInicial é anterior à data de criação da recorrência.

• O location referenciado por rec.loc inexiste.

• O location referenciado por rec.loc já está sendo utilizado por outra recorrência.

• O campo rec.status não respeita o schema.

• A recorrência encontra-se encerrada.

• O campo rec.loc somente pode ser alterado quando a recorrência apresentar-se com o status CRIADA.

• O campo rec.calendario.dataInicial somente pode ser alterado quando a recorrência apresentar-se com o status CRIADA.

• O campo rec.dadosJornada.txid não pode ser alterado quando a recorrência apresentar-se com o status REJEITADA ou CANCELADA.

RecConsultaInvalida

• Significado: os parâmetros de consulta à lista de recorrências que não respeitam o schema ou não fazem sentido semanticamente.

• HTTP Status Code: 400.

• endpoints:

• endpoints: GET /rec e GET /rec/{idRec}.

Violações específicas para o endpoint GET /rec:

• algum dos parâmetros informados para a consulta não respeita o schema.

• o timestamp representado pelo parâmetro fim é anterior ao timestamp representado pelo parâmetro inicio.

• ambos os parâmetros cpf e cnpj estão preenchidos.

• o parâmetro paginacao.paginaAtual é negativo.

• o parâmetro paginacao.itensPorPagina é negativo.

Violações específicas para o endpoint GET /rec/{idRec}:

• o parâmetro txid não corresponde a uma cobrança compatível com o campo ativacao.tipoJornada. (Exemplo: txid correspondente a uma CobV e ativação.tipoJornada igual a JORNADA_3.)

• o parâmetro txid corresponde a uma cobrança imediata diferente da informada no campo ativação.dadosJornada.txid.

4.3.3 SolicRec

Esta seção reúne erros retornados pelos endpoints organizados sob a tag SolicRec. Esses erros indicam problemas no gerenciamento de uma solicitação de confirmação de recorrência. SolicRecNaoEncontrada

• Significado: Solicitação de recorrência não encontrada para o idSolicRec informado.

• HTTP Status Code: 404.

• endpoints: [GET] /solicrec/{idSolicRec}.

SolicRecOperacaoInvalida

• Significado: a requisição que busca criar ou alterar uma solicitação de confirmação de recorrência não respeita o schema ou está semanticamente errada.

• HTTP Status Code: 400.

• endpoints: [POST] /solicrec e PATCH /solicrec/{idSolicRec}.

Violações para o endpoint POST /solicrec:

• O objeto solicrec.calendario não respeita o schema.

• O campo solicrec.calendario.dataExpiracaoSolicitacao é anterior à data de criação da solicitação da recorrência.

• O objeto solicrec.destinatario não respeita o schema.

• Existe uma solicitação ativa referente ao mesmo solicrec.idRec.

Violações para o endpoint PATCH /solicrec/{idSolicRec}:

• Não é possível cancelar uma solicitação de recorrência com o status diferente de CRIADA

ENVIADA ou RECEBIDA.

4.3.4 CobR

Esta seção reúne erros retornados pelos endpoints organizados sob a tag CobR. Esses erros indicam problemas no gerenciamento de uma cobrança recorrente.

CobRNaoEncontrado

• Significado: Cobrança não encontrada para o txid informado.

• HTTP Status Code: 404.

• endpoints: [GET|PATCH] /cobr/{txid} e [POST] /cobr/{txid}/retentativa/{data}.

CobROperacaoInvalida

• Significado: a requisição que busca alterar ou criar uma cobrança recorrente não respeita o schema ou está semanticamente errada.

• HTTP Status Code: 400.

• endpoints: [POST|PUT|PATCH] /cobr/{txid} e [POST] /cobr/{txid}/retentativa/{data}.

Violações para o endpoint POST|PUT /cobr/{txid}:

• O campo cobr.infoAdicional não respeita o schema.

• O campo cobr.status não respeita o schema.

• O objeto cobr.calendario não respeita o schema.

• O campo cobr.calendario.dataDeVencimento é anterior à data de criação da cobrança.

• O campo cobr.valor não respeita o schema.

O objeto cobr.recebedor não respeita o schema.

Os campos cobr.recebedor.conta e cobr.recebedor.agencia correspondem a uma conta que não pertence a este usuário recebedor.

• O objeto cobr.devedor não respeita o schema.

• O campo cobr.txid encontra-se em uso.

• Existe uma CobR com status diferente de REJEITADA e CANCELADA referente ao mesmo cobr.idRec com calendario.dataDeVencimento no mesmo ciclo.

Violações para o endpoint PATCH /cobr/{txid}:

• Não é possível cancelar uma cobrança em uma data igual ou maior que a data prevista da primeira tentativa de liquidação.

Violações para o endpoint POST /cobr/{txid}/retentativa/{data}:

• Existe uma tentativa com status SOLICITADA ou AGENDADA.

• Existe uma tentativa em andamento.

• Existe uma tentativa ativa.

• Existe uma tentativa não finalizada.

• Existe uma tentativa vigente para a data informada.

• O parâmetro data não corresponde a uma data futura.

• A política configurada na recorrência não permite retentativa de cobrança.

CobRConsultaInvalida

• Significado: os parâmetros de consulta à lista de cobranças que não respeitam o schema ou não fazem sentido semanticamente.

• HTTP Status Code: 400.

• endpoints: GET /cobr e GET /cobr/{txid}.

Violações específicas para o endpoint GET /cobr:

• algum dos parâmetros informados para a consulta não respeita o schema.

• o timestamp representado pelo parâmetro fim é anterior ao timestamp representado pelo

parâmetro inicio.

• ambos os parâmetros cpf e cnpj estão preenchidos.

• o parâmetro paginacao.paginaAtual é negativo.

• o parâmetro paginacao.itensPorPagina é negativo.

4.3.5 Cob

Esta seção reúne erros retornados pelos endpoints organizados sob a tag **Cob**. Esses erros indicam problemas no gerenciamento de uma cobrança para pagamento imediato.

CobNaoEncontrado

• Significado: Cobrança não encontrada para o txid informado.

• HTTP Status Code: 404.

• endpoints: [GET|PATCH] /cob/{txid}.

CobOperacaoInvalida

• Significado: a requisição que busca alterar ou criar uma cobrança para pagamento imediato não respeita o schema ou está semanticamente errada.

HTTP Status Code: 400.

• endpoints: [POST|PUT|PATCH] /cob/{txid}.

Violações para os endpoints PUT|PATCH /cob/{txid}:

• O campo cob.calendario.expiracao é igual ou menor que zero.

• O campo cob.valor.original não respeita o schema.

• O campo cob.valor.original é zero.

• O objeto cob.devedor não respeita o schema.

• O campo cob.chave não respeita o schema.

• O campo cob.chave corresponde a uma conta que não pertence a este usuário recebedor.

• O campo solicitacaoPagador não respeita o schema.

• O objeto infoAdicionais não respeita o schema.

• O location referenciado por loc.id inexiste.

• O location referenciado por loc.id já está sendo utilizado por outra cobrança.

• O location referenciado por cob.loc.id apresenta tipo "cobv" (deveria ser "cob").

Violações específicas para o endpoint PUT /cob/{txid}:

• A cobrança já existe, não está no status ATIVA, e a presente requisição busca alterá-la.

Violações específicas para o endpoint PATCH /cob/{txid}:

• A cobrança não está ATIVA, e a presente requisição busca alterá-la.

• A cobrança está ATIVA, e a presente requisição propõe alterar seu status para REMOVIDA_PELO_USUARIO_RECEBEDOR juntamente com outras alterações (não faz sentido remover uma cobrança ao mesmo tempo em que se realizam alterações que não serão aproveitadas).

• o campo cob.status não respeita o schema.

CobConsultaInvalida

• Significado: os parâmetros de consulta à lista de cobranças para pagamento imediato não respeitam o schema ou não fazem sentido semanticamente.

• HTTP Status Code: 400.

• endpoints: GET /cob e GET /cob/{txid}.

Violações específicas para o endpoint GET /cob:

• algum dos parâmetros informados para a consulta não respeita o schema.

• o timestamp representado pelo parâmetro fim é anterior ao timestamp representado pelo parâmetro inicio.

• ambos os parâmetros cpf e cnpj estão preenchidos.

• o parâmetro paginacao.paginaAtual é negativo.

• o parâmetro paginacao.itensPorPagina é negativo.

Violações específicas para o endpoint GET /cob/{txid}:

o parâmetro revisao corresponde a uma revisão inexistente para a cobrança apontada pelo parâmetro txid.

4.3.6 CobV

Esta seção reúne erros retornados pelos endpoints organizados sob a tag CobV. Esses erros indicam problemas no gerenciamento de uma cobrança com vencimento.

CobVNaoEncontrada

• Significado: Cobrança com vencimento não encontrada para o txid informado.

• HTTP Status Code: 404.

• endpoints: [GET|PATCH] /cobv/{txid}.

CobVOperacaoInvalida

• Significado: a requisição que busca alterar ou criar uma cobrança com vencimento não respeita o schema ou está semanticamente errada.

• HTTP Status Code: 400.

• endpoints: [PUT|PATCH] /cobv/{txid}.

Violações para os endpoints PUT|PATCH /cobv/{txid}:

• Este txid está associado a um lote e no referido lote, o status desta cobrança está atribuído como "EM_PROCESSAMENTO" ou "NEGADA".

• O campo cobv.calendario.dataDeVencimento é anterior à data de criação da cobrança.

• O campo cobv.calendario.validadeAposVencimento é menor do que zero.

• O objeto cobv.devedor não respeita o schema.

• O objeto cobv.devedor não respeita o schema.

• O campo cobv.chave não respeita o schema.

• O campo cobv.chave corresponde a uma conta que não pertence a este usuário recebedor.

• O campo solicitacaoPagador não respeita o schema.

• O objeto infoAdicionais não respeita o schema.

• O location referenciado por cobv.loc.id inexiste.

• O location referenciado por cobv.loc.id já está sendo utilizado por outra cobrança.

• O location referenciado por cobv.loc.id apresenta tipo "cob" (deveria ser "cobv").

• O campo cobv.valor.original não respeita o schema.

• O campo cobv.valor.original apresenta o valor zero.

• O objeto cobv.valor.multa não respeita o schema.

• O objeto cobv.valor.juros não respeita o schema.

• O objeto cobv.valor.abatimento não respeita o schema.

• O objeto cobv.valor.desconto não respeita o schema.

• O objeto cobv.valor.abatimento representa um valor maior ou igual ao valor da cobrança original ou maior ou igual a 100%.

• O objeto cobv.valor.desconto apresenta algum elemento de desconto que representa um valor maior ou igual ao valor da cobrança original ou maior ou igual a 100%.

• O objeto cobv.valor.desconto apresenta algum elemento cuja data seja posterior à data de vencimento representada por calendario.dataDeVencimento.

• O objeto cobv.valor.desconto apresenta modalidade no porém cobv.valor.desconto.valorPerc encontra-se preenchido valor 1 ou 2,

• O objeto cobv.valor.desconto apresenta modalidade no valor 1 ou 2, array cobv.valor.desconto.descontoDataFixa está vazio ou nulo. porém o

• O objeto cobv.valor.desconto apresenta modalidade nos valores de 3 a 6, elemento cobv.valor.desconto.valorPerc não está preenchido. porém o

• O objeto cobv.valor.desconto apresenta modalidade nos valores de 3 a 6, elemento cobv.valor.desconto.descontoDataFixa está preenchido ou não nulo. porém o

Violações específicas para o endpoint PUT /cobv/{txid}:

• A cobrança já existe, não está ATIVA, e a presente requisição busca alterá-la

Violações específicas para o endpoint PATCH /cobv/{txid}:

• A cobrança não está ATIVA, e a presente requisição busca alterá-la

• A cobrança está ATIVA, e a presente requisição propõe alterar seu status para REMOVIDA_PELO_USUARIO_RECEBEDOR juntamente com outras alterações (não faz sentido remover uma cobrança ao mesmo tempo em que se realizam alterações que não serão aproveitadas).

• o campo cob.status não respeita o schema.

CobVConsultaInvalida

• Significado: os parâmetros de consulta à lista de cobranças com vencimento não respeitam o schema ou não fazem sentido semanticamente.

• HTTP Status Code: 400.

• endpoints: GET /cobv e GET /cobv/{txid}.

Violações específicas para o endpoint GET /cobv:

• algum dos parâmetros informados para a consulta não respeita o schema.

• o timestamp representado pelo parâmetro fim é anterior ao timestamp representado pelo parâmetro inicio.

• ambos os parâmetros cpf e cnpj estão preenchidos.

• o parâmetro paginacao.paginaAtual é negativo.

• o parâmetro paginacao.itensPorPagina é negativo.

Violações específicas para o endpoint GET /cobv/{txid}:

• o parâmetro revisao corresponde a uma revisão inexistente para a cobrança apontada pelo parâmetro txid.

Referências

Para o desenvolvimento da API, deverão ser seguidas as orientações do GitHub do Bacen, cujos link disponibilizamos:

https://github.com/bacen/pix-api.

Swagger UI


CAIXA